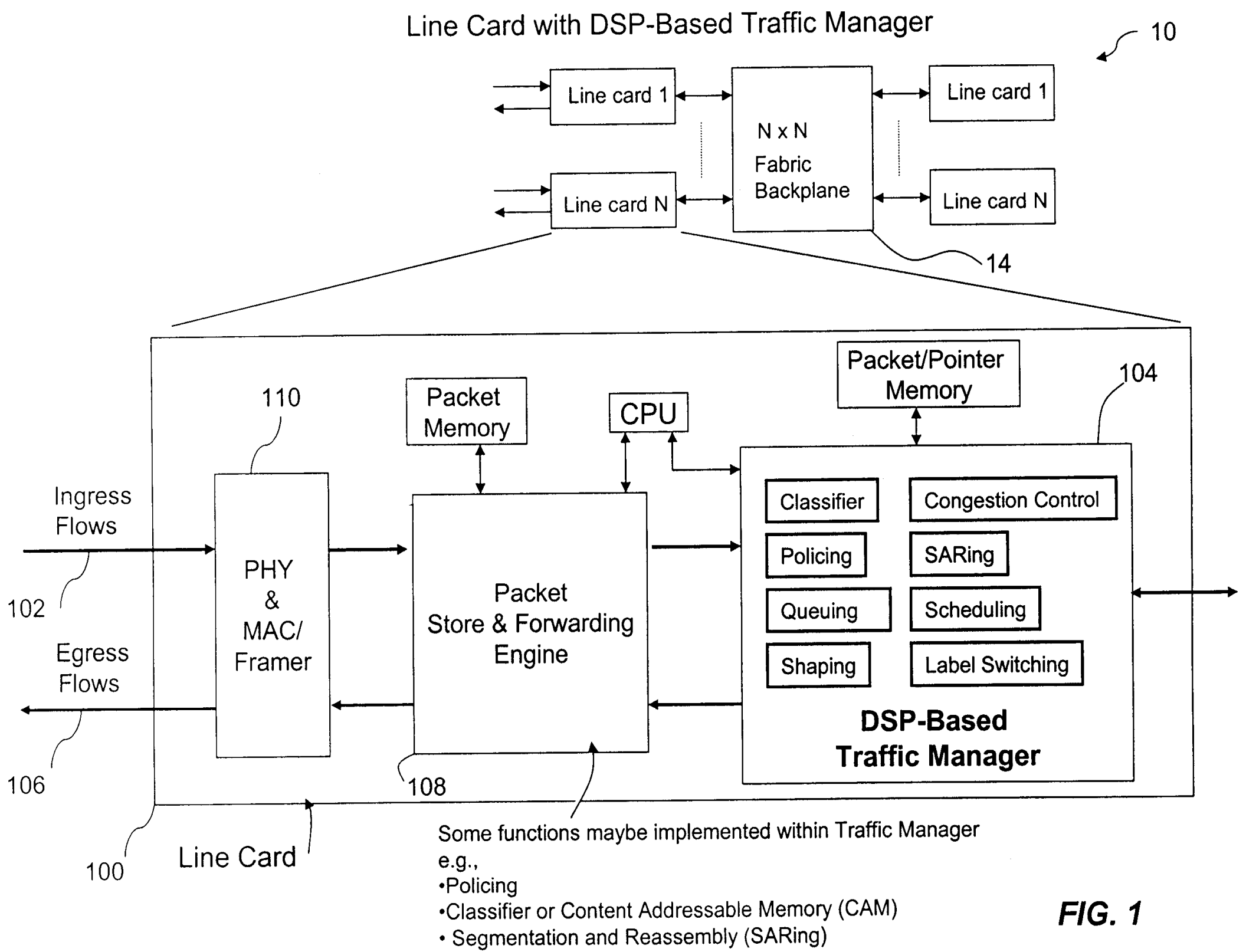

Line Card with DSP-Based Traffic Manager
10
Line card 1
Line card N
N x N
Fabric
Backplane
Line card 1
Line card N
14
Packet/Pointer
Memory
104
110
Packet
Memory
CPU
Ingress
Flows
102
Egress
Flows
106
PHY
&
MAC/
Framer
Packet
Store & Forwarding
Engine
108
Classifier
Congestion Control
Policing
SARing
Queuing
Scheduling
Shaping
Label Switching
DSP-Based
Traffic Manager
100
Line Card
Some functions maybe implemented within Traffic Manager
e.g.,
•Policing
•Classifier or Content Addressable Memory (CAM)
• Segmentation and Reassembly (SARing)

FIG. 1

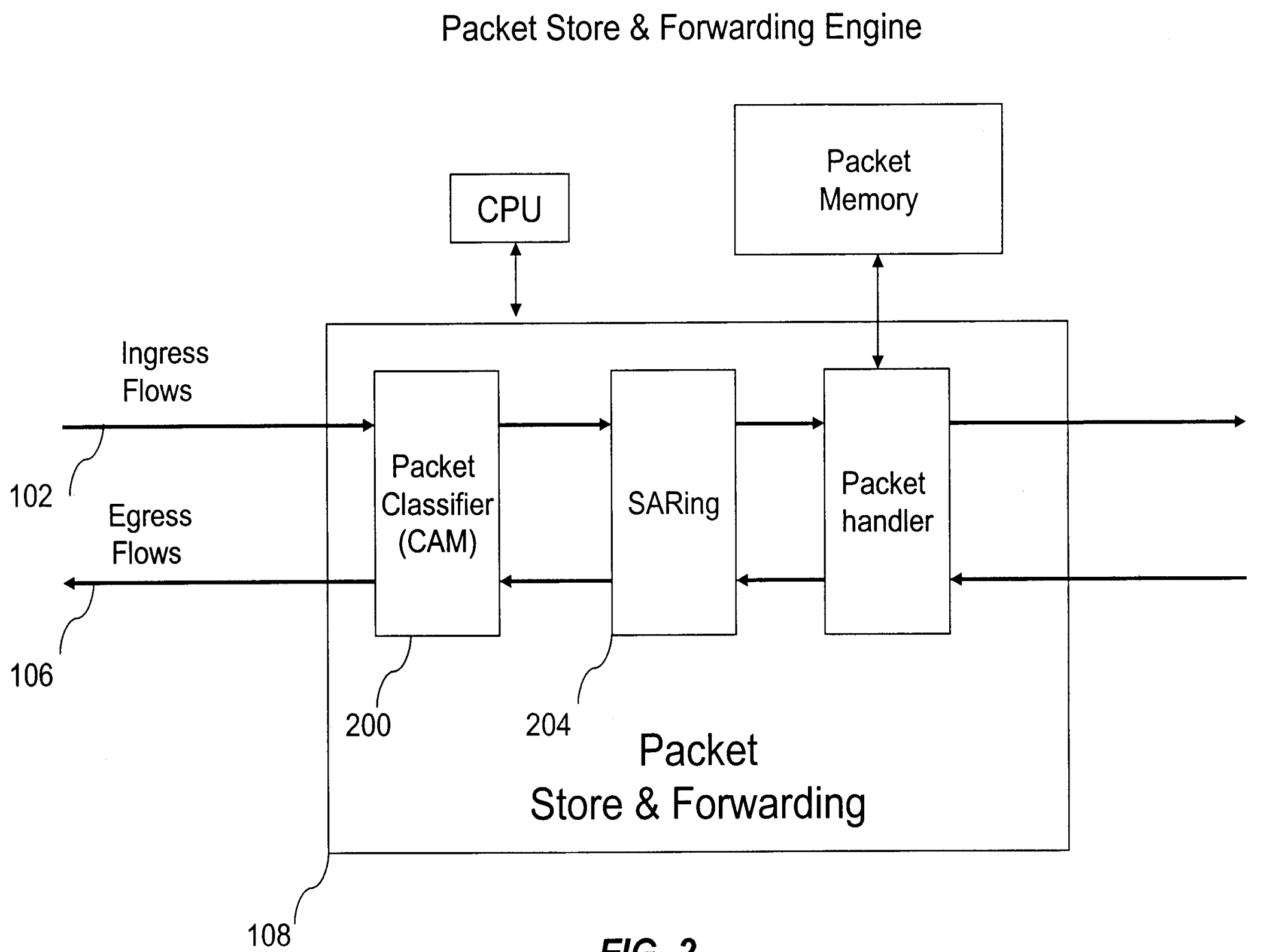

Packet Store & Forwarding Engine
CPU
Packet Memory
Ingress Flows
Egress Flows
102
106
Packet Classifier (CAM)
SARing
Packet handler
200
204
Packet Store & Forwarding
108

FIG. 2

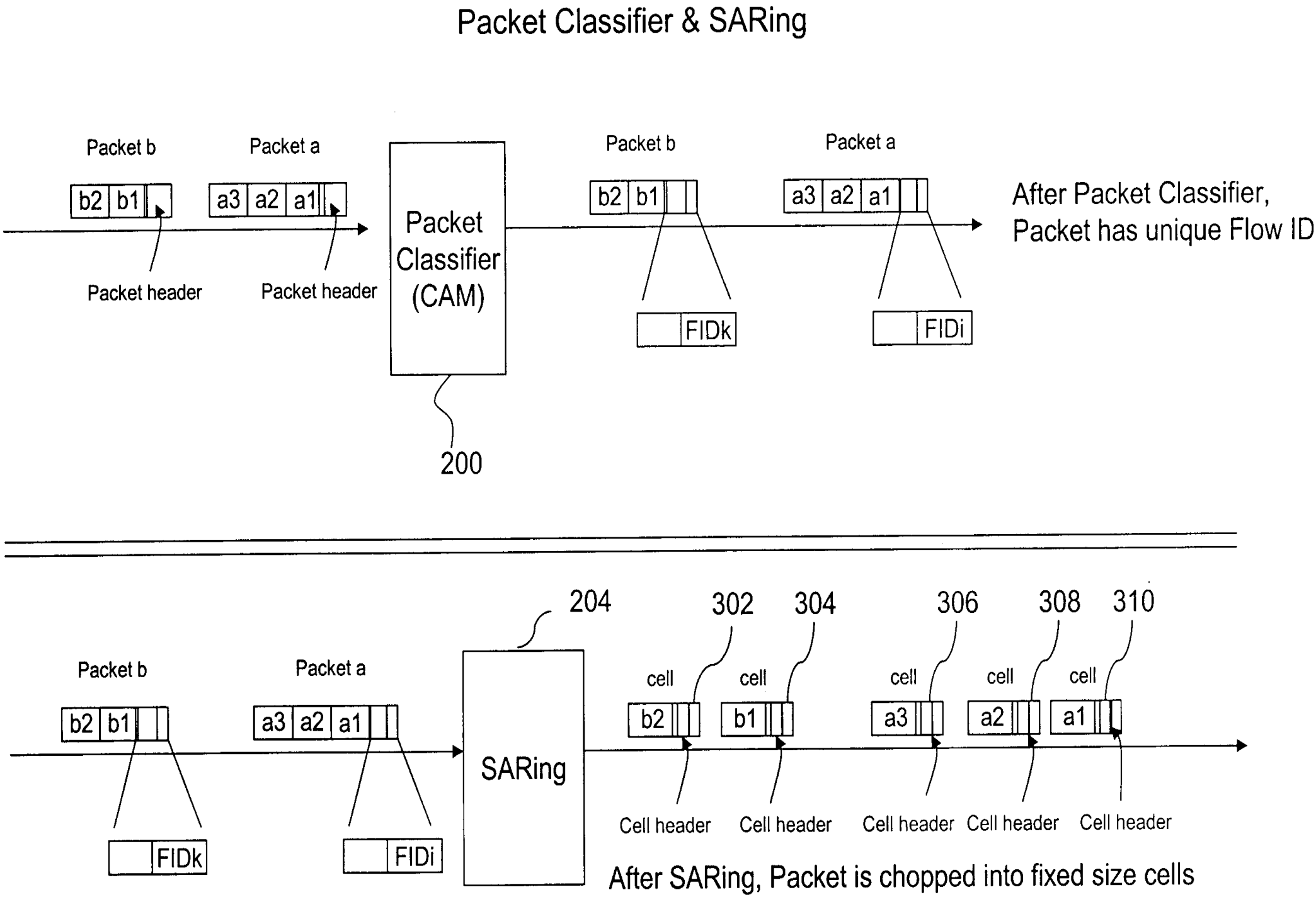

Packet Classifier & SARing
Packet b
Packet a
b2 b1
a3 a2 a1
Packet header
Packet header
Packet Classifier (CAM)
200
Packet b
Packet a
b2 b1
a3 a2 a1
FIDk
FIDi
After Packet Classifier, Packet has unique Flow ID
204
302
304
306
308
310
Packet b
Packet a
b2 b1
a3 a2 a1
FIDk
FIDi
SARing
cell
cell
cell
cell
cell
b2
b1
a3
a2
a1
Cell header
Cell header
Cell header
Cell header
Cell header
After SARing, Packet is chopped into fixed size cells

FIG. 3

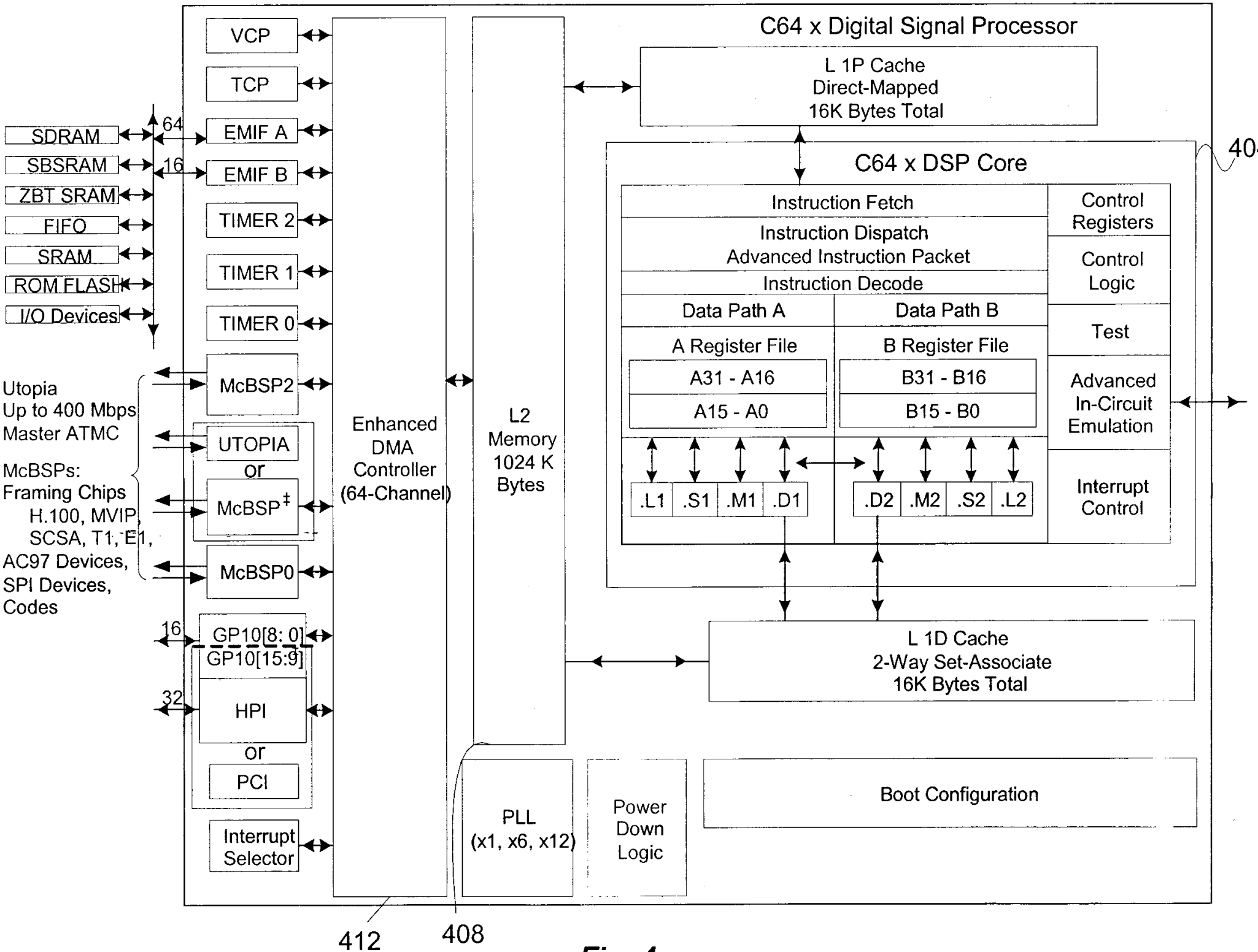

C64 x Digital Signal Processor
L 1P Cache
Direct-Mapped
16K Bytes Total
404
C64 x DSP Core
Instruction Fetch
Instruction Dispatch
Advanced Instruction Packet
Instruction Decode
Data Path A
Data Path B
A Register File
B Register File
A31 - A16
A15 - A0
B31 - B16
B15 - B0
.L1
.S1
.M1
.D1
.D2
.M2
.S2
.L2
Control Registers
Control Logic
Test
Advanced In-Circuit Emulation
Interrupt Control
L 1D Cache
2-Way Set-Associate
16K Bytes Total
VCP
TCP
EMIF A
EMIF B
TIMER 2
TIMER 1
TIMER 0
McBSP2
UTOPIA
or
McBSP‡
McBSP0
GP10[8: 0]
GP10[15:9]
HPI
or
PCI
Interrupt Selector
64
16
32
SDRAM
SBSRAM
ZBT SRAM
FIFO
SRAM
ROM FLASH
I/O Devices
Utopia
Up to 400 Mbps
Master ATMC
McBSPs:
Framing Chips
H.100, MVIP,
SCSA, T1, E1,
AC97 Devices,
SPI Devices,
Codes
Enhanced DMA Controller (64-Channel)
L2 Memory 1024 K Bytes
PLL (x1, x6, x12)
Power Down Logic
Boot Configuration
412
408

Fig. 4

FUNCTIONAL BLOCK DIAGRAM

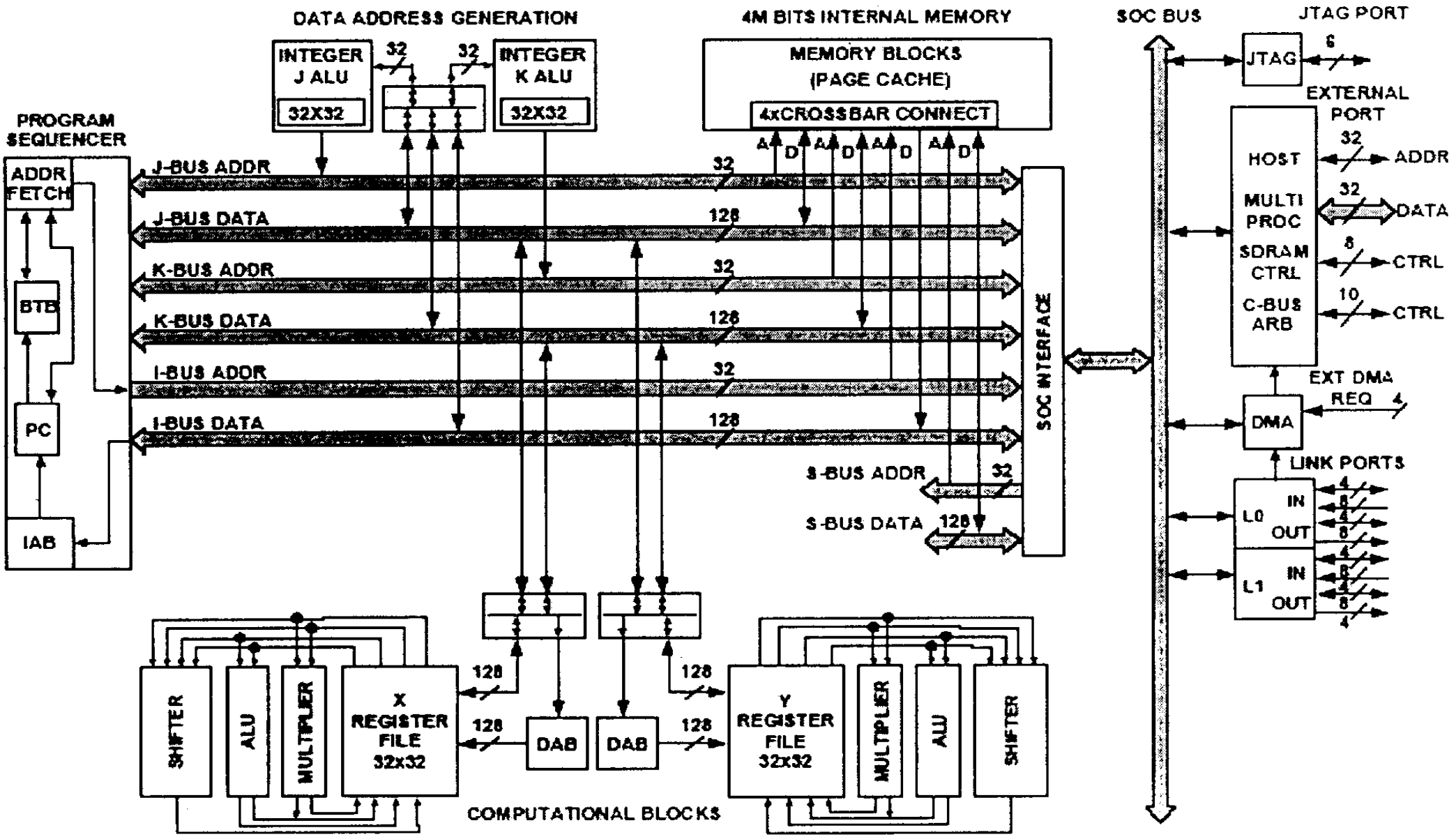


Fig. 5

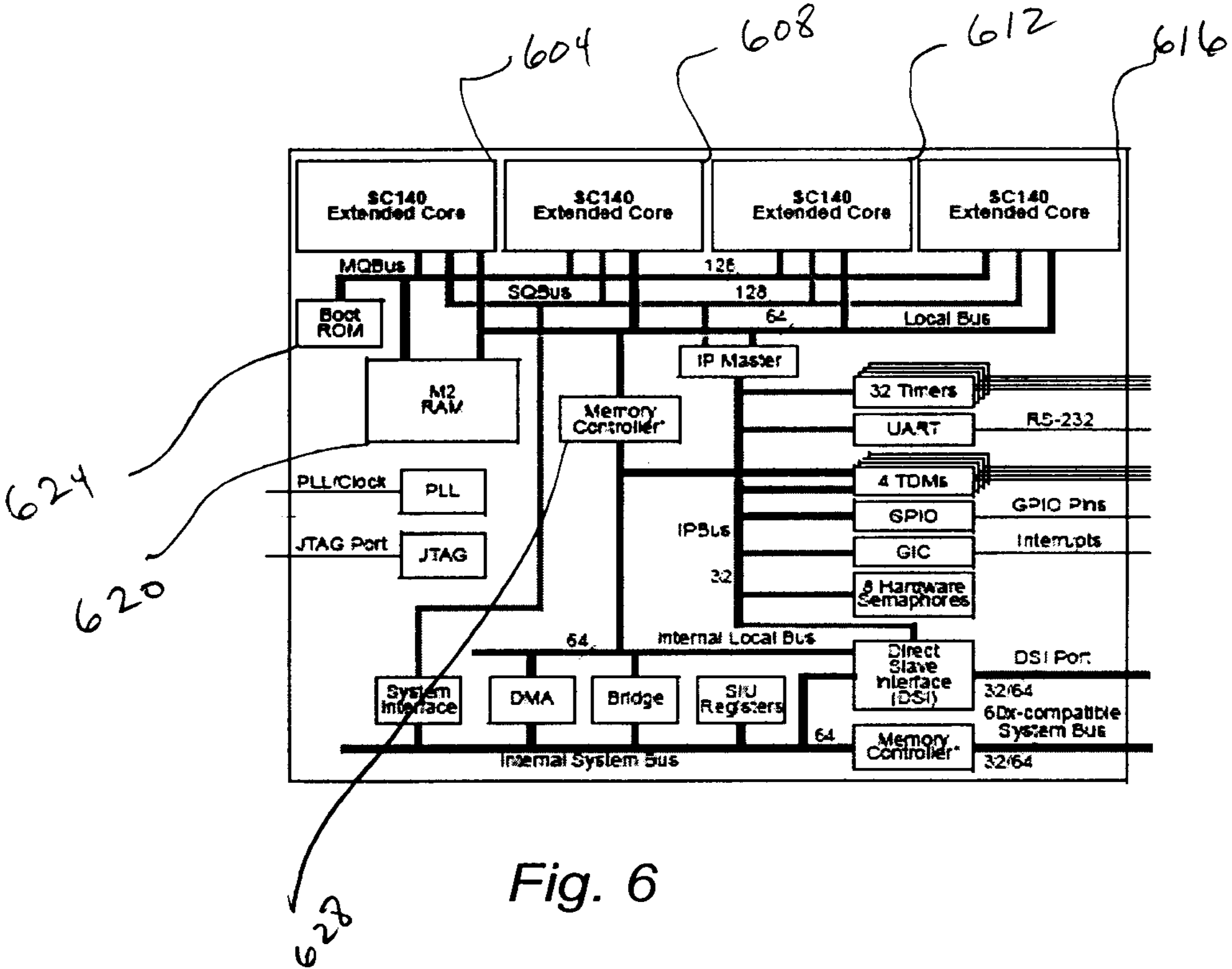

604
608
612
616
SC140 Extended Core
SC140 Extended Core
SC140 Extended Core
SC140 Extended Core
MQBus
SQBus
128
128
64
Local Bus
Boot ROM
M2 RAM
Memory Controller
IP Master
32 Timers
UART
RS-232
4 TDMs
GPIO
GPIO Pins
GIC
Interrupts
8 Hardware Semaphores
IPBus
32
PLL/Clock
PLL
JTAG Port
JTAG
64
Internal Local Bus
Direct Slave Interface (DSI)
DSI Port
32/64
System Interface
DMA
Bridge
SIU Registers
64
Memory Controller
6Dx-compatible System Bus
32/64
Internal System Bus
624
620
628

Fig. 6

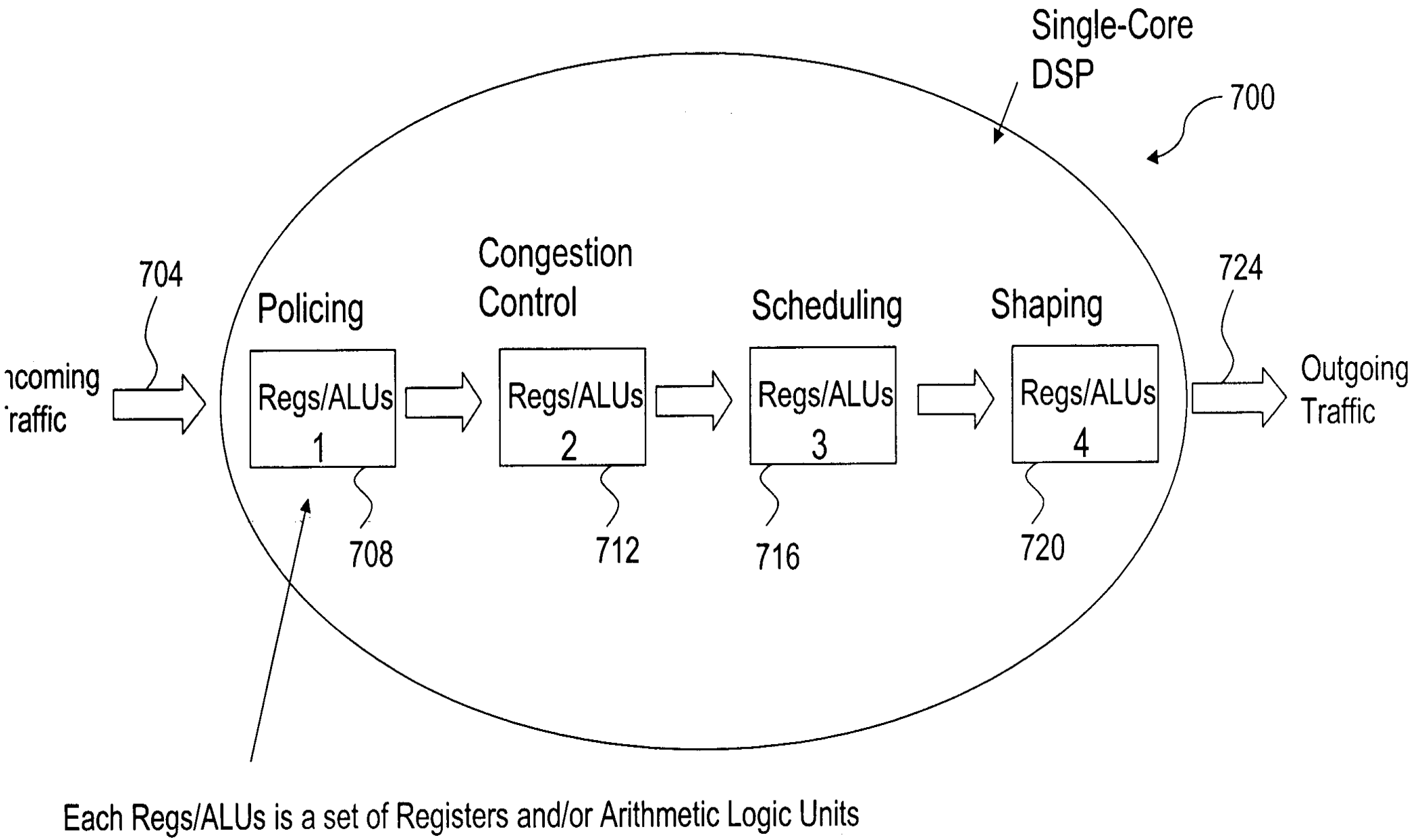

Implement Traffic Management Functions in
Single-Core DSP by Pipeline Processing Approach
Single-Core
DSP
700
704
Policing
Congestion
Control
Scheduling
Shaping
724
ıcoming
raffic
Regs/ALUs
1
Regs/ALUs
2
Regs/ALUs
3
Regs/ALUs
4
Outgoing
Traffic
708
712
716
720
Each Regs/ALUs is a set of Registers and/or Arithmetic Logic Units

FIG. 7

Implement Traffic Management Functions in
Single-Core DSP by Parallel Processing Approach

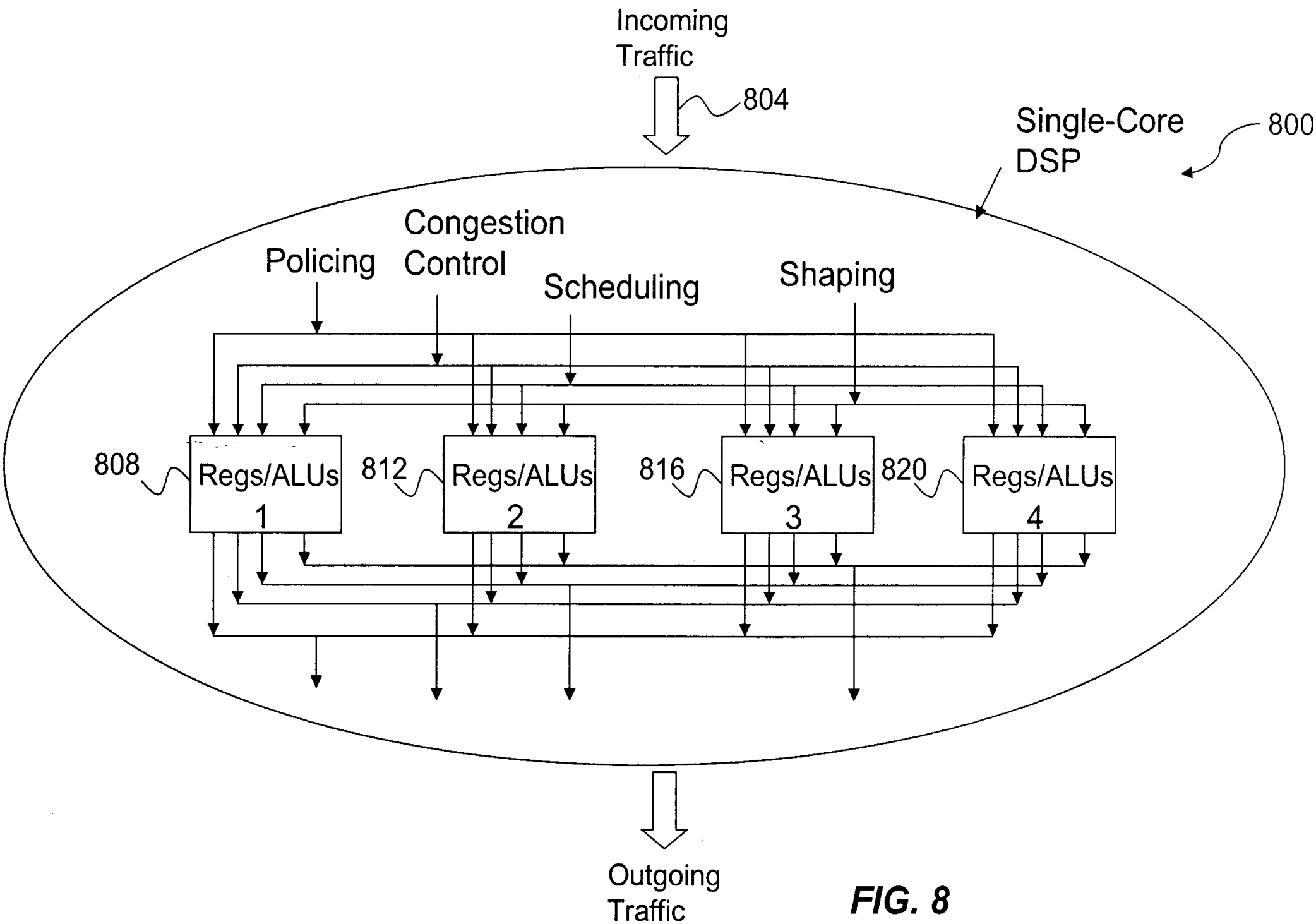


FIG. 8

Implement Traffic Management Functions in Single-Core DSP
by Mixing Pipeline and Parallel Processing Approach

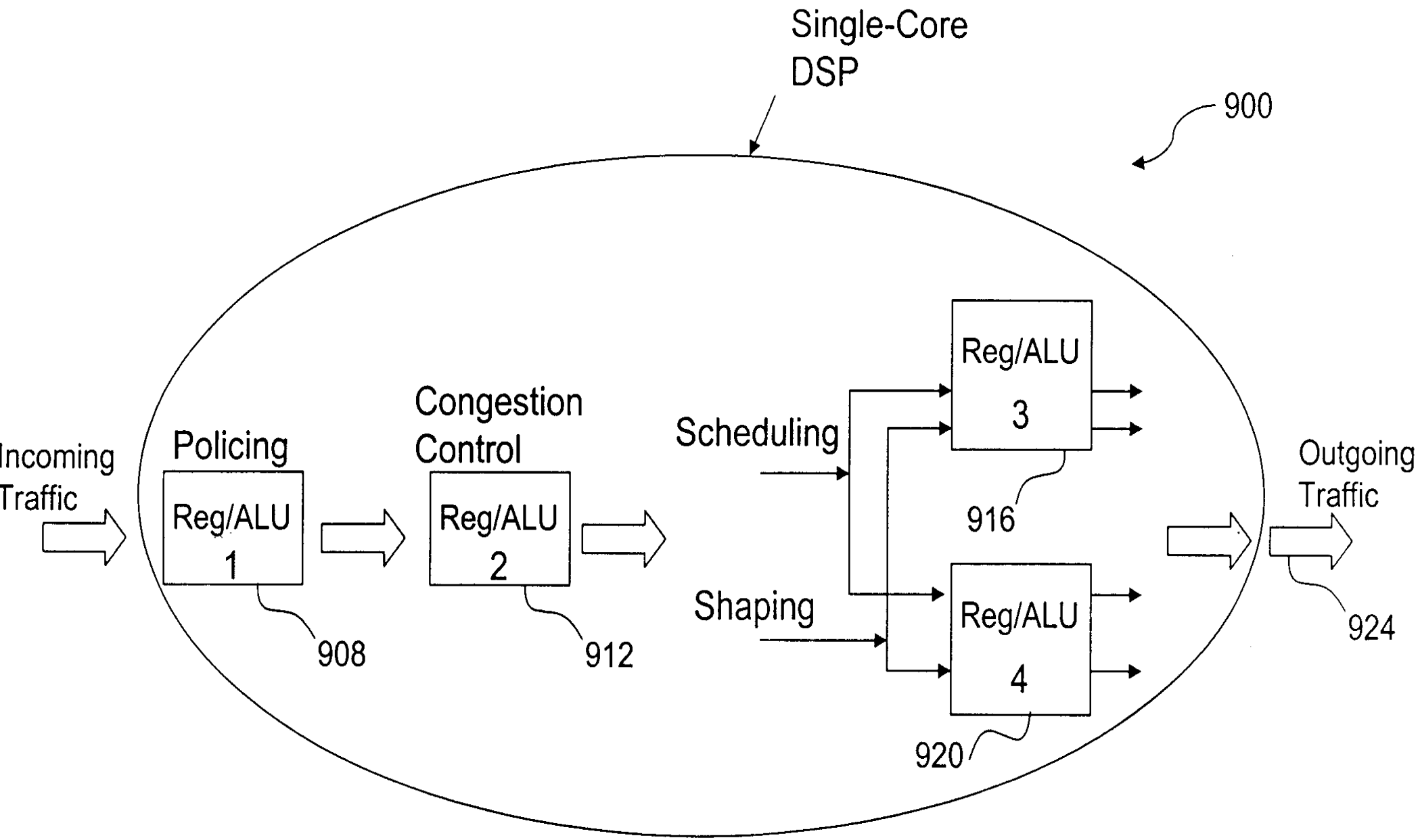


FIG. 9

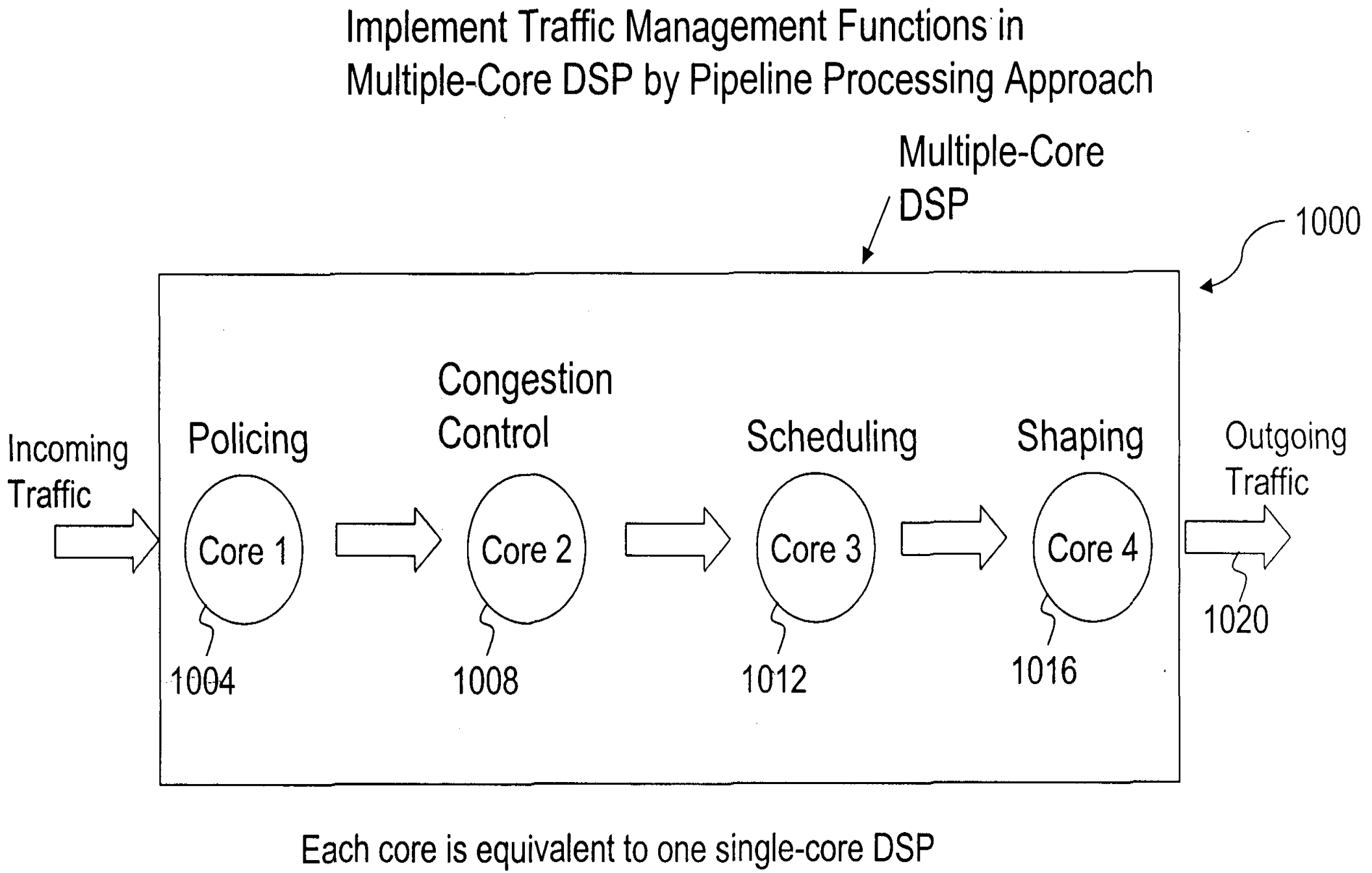

Implement Traffic Management Functions in
Multiple-Core DSP by Pipeline Processing Approach
Multiple-Core
DSP
1000
Incoming
Traffic
Policing
Congestion
Control
Scheduling
Shaping
Outgoing
Traffic
Core 1
Core 2
Core 3
Core 4
1004
1008
1012
1016
1020
Each core is equivalent to one single-core DSP

FIG. 10

Implement Traffic Management Functions in Multiple-Core DSP by Parallel Processing Approach

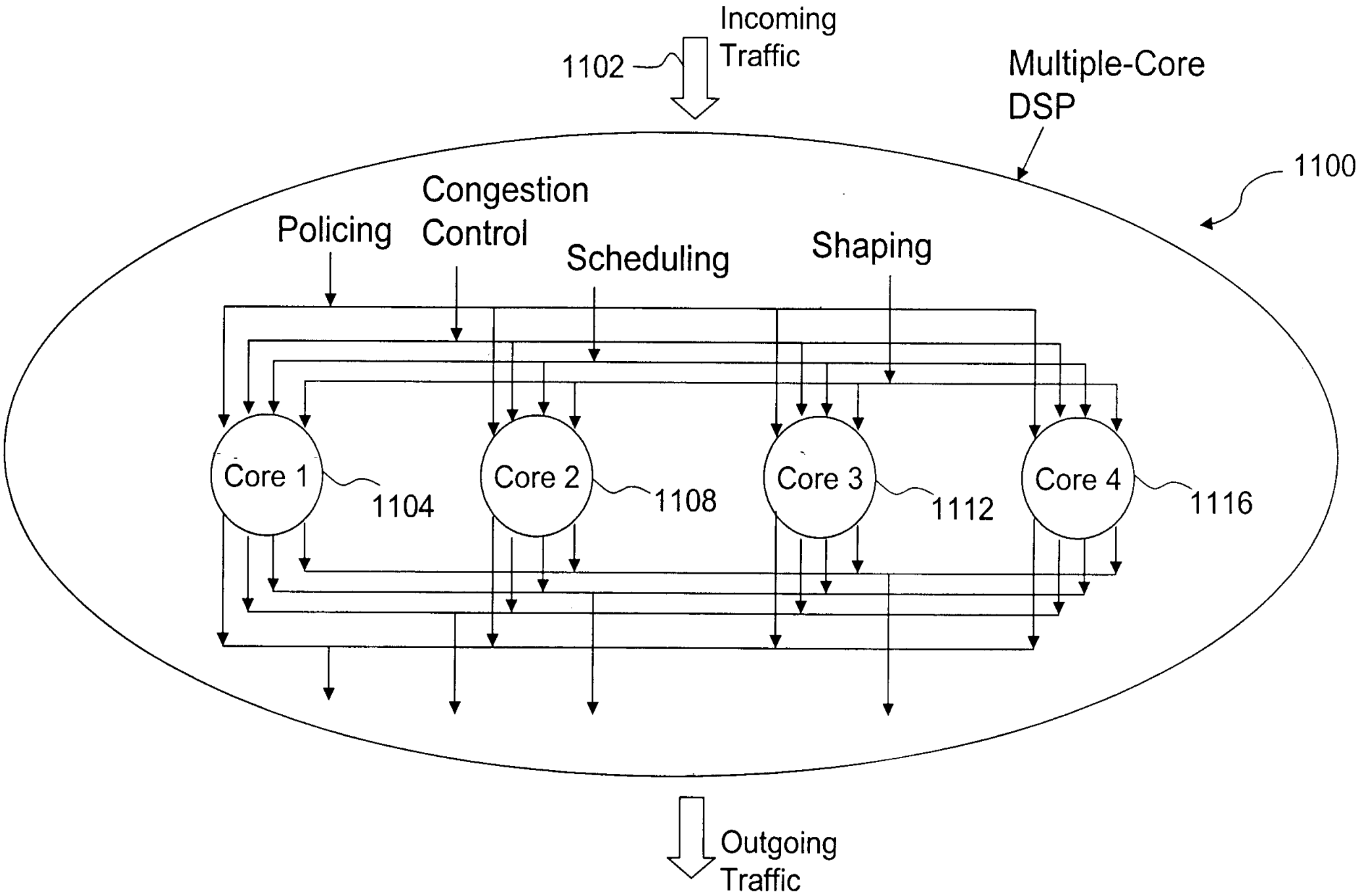


FIG. 11

Implement Traffic Management Functions in Multiple-Core DSP by Mixing Pipeline and Parallel Processing Approach

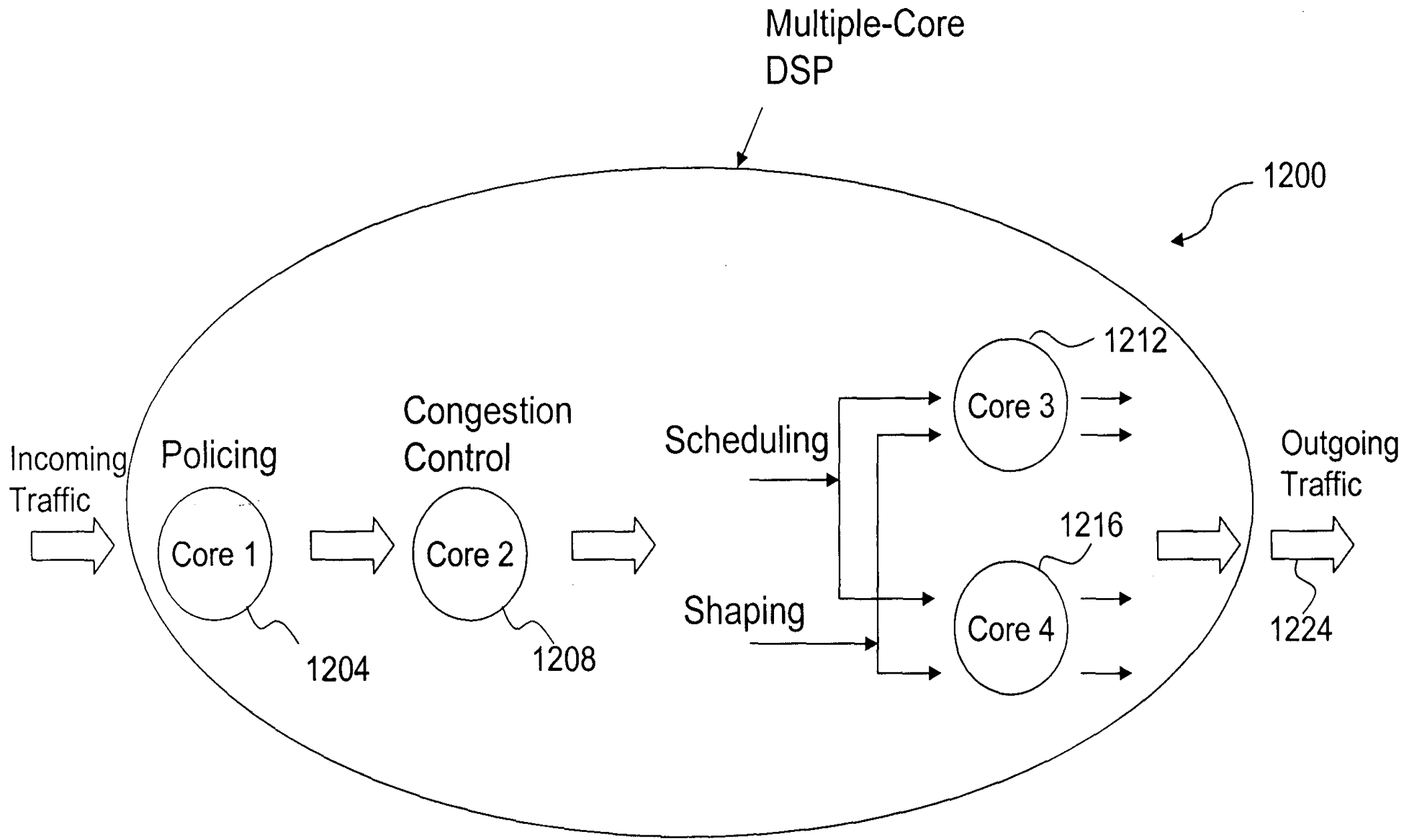


FIG. 12

Using Mailboxes to Communicate between Different DSP Cores

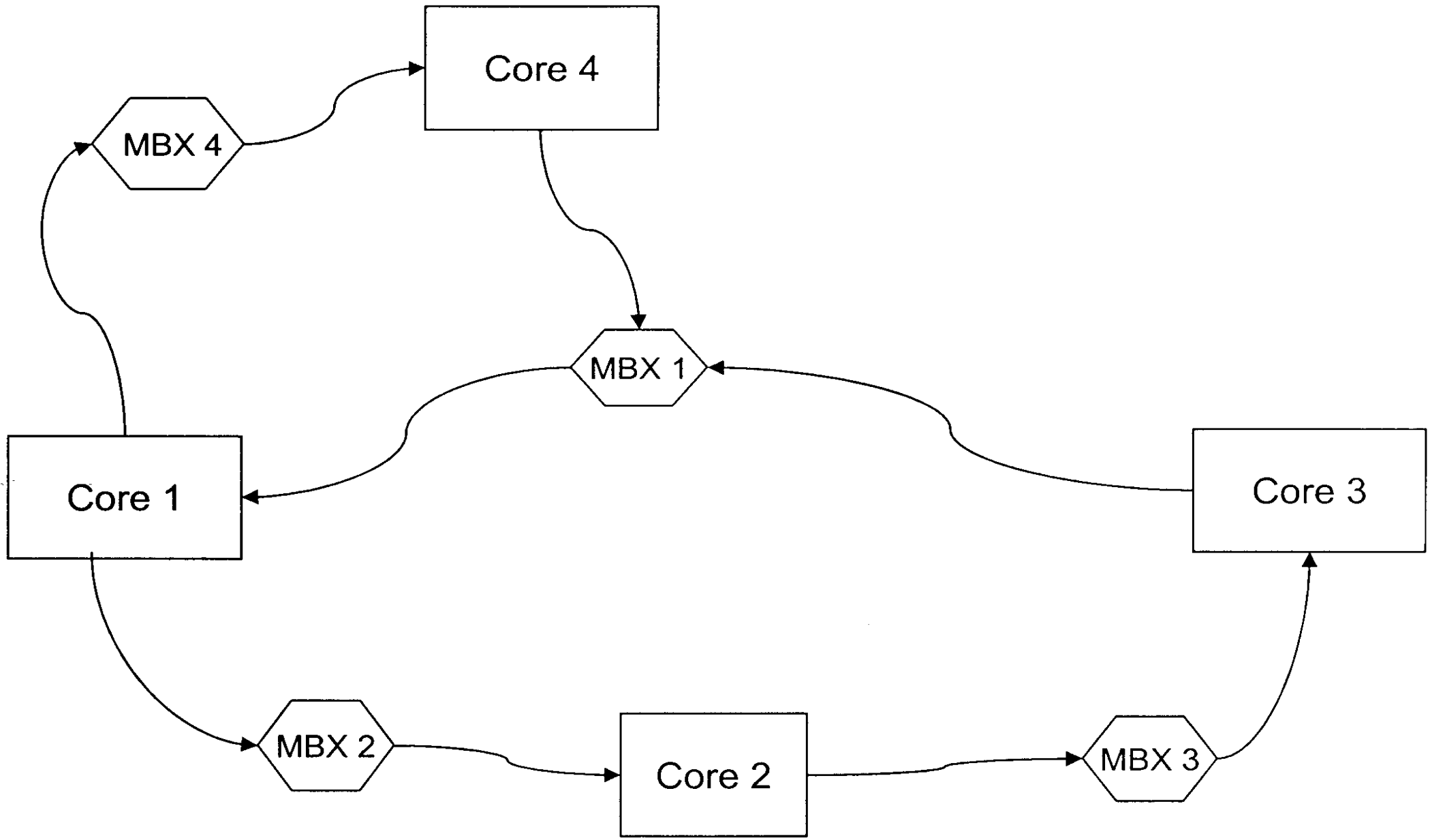


FIG. 13

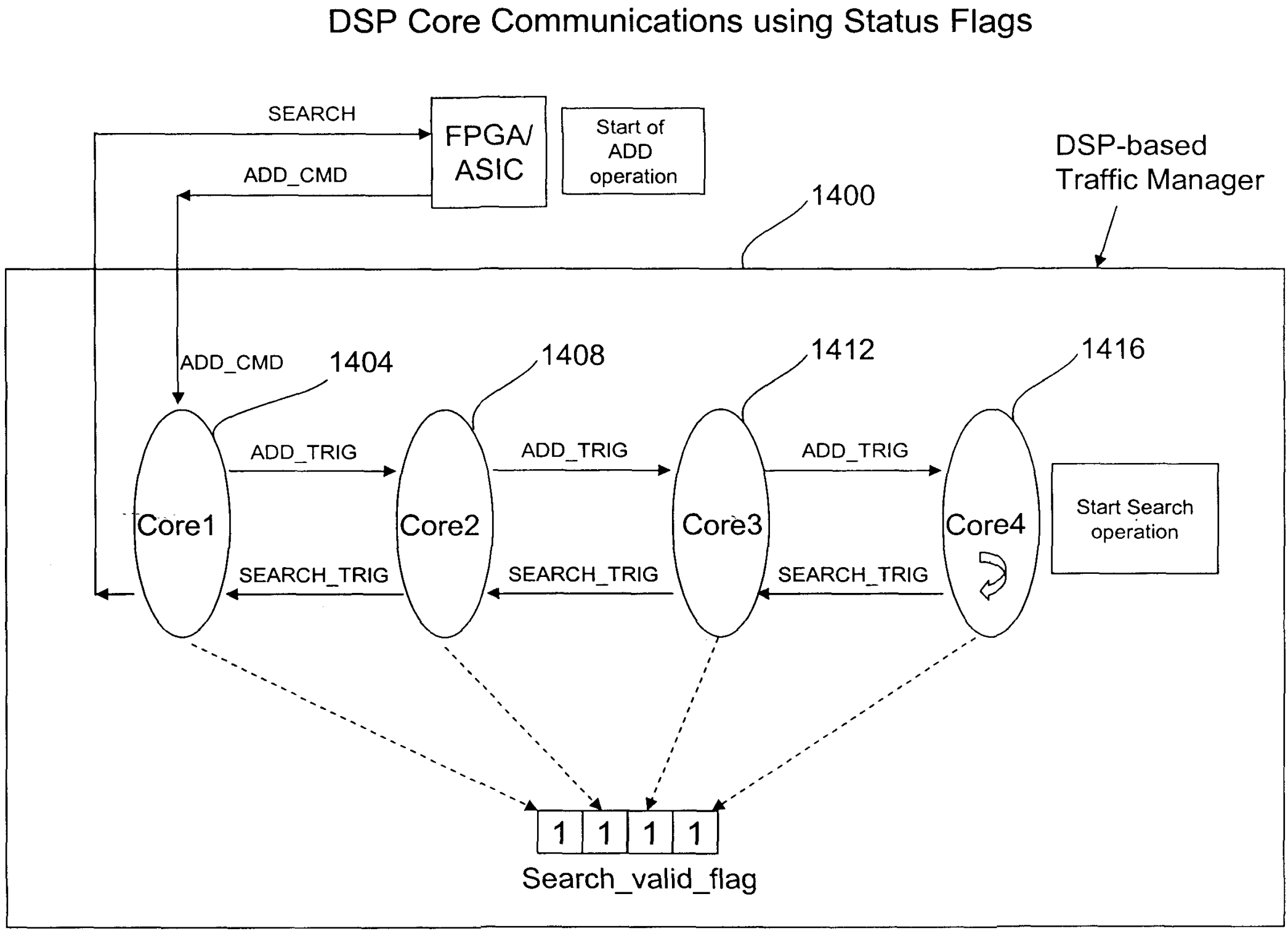

DSP Core Communications using Status Flags
SEARCH
ADD_CMD
FPGA/ ASIC
Start of ADD operation
DSP-based Traffic Manager
1400
ADD_CMD
1404
1408
1412
1416
ADD_TRIG
ADD_TRIG
ADD_TRIG
Core1
Core2
Core3
Core4
Start Search operation
SEARCH_TRIG
SEARCH_TRIG
SEARCH_TRIG
1 1 1 1
Search_valid_flag

FIG. 14

Use Sync_Pattern to Guarantee Multiple DSP Cores Enter Wait_Loop

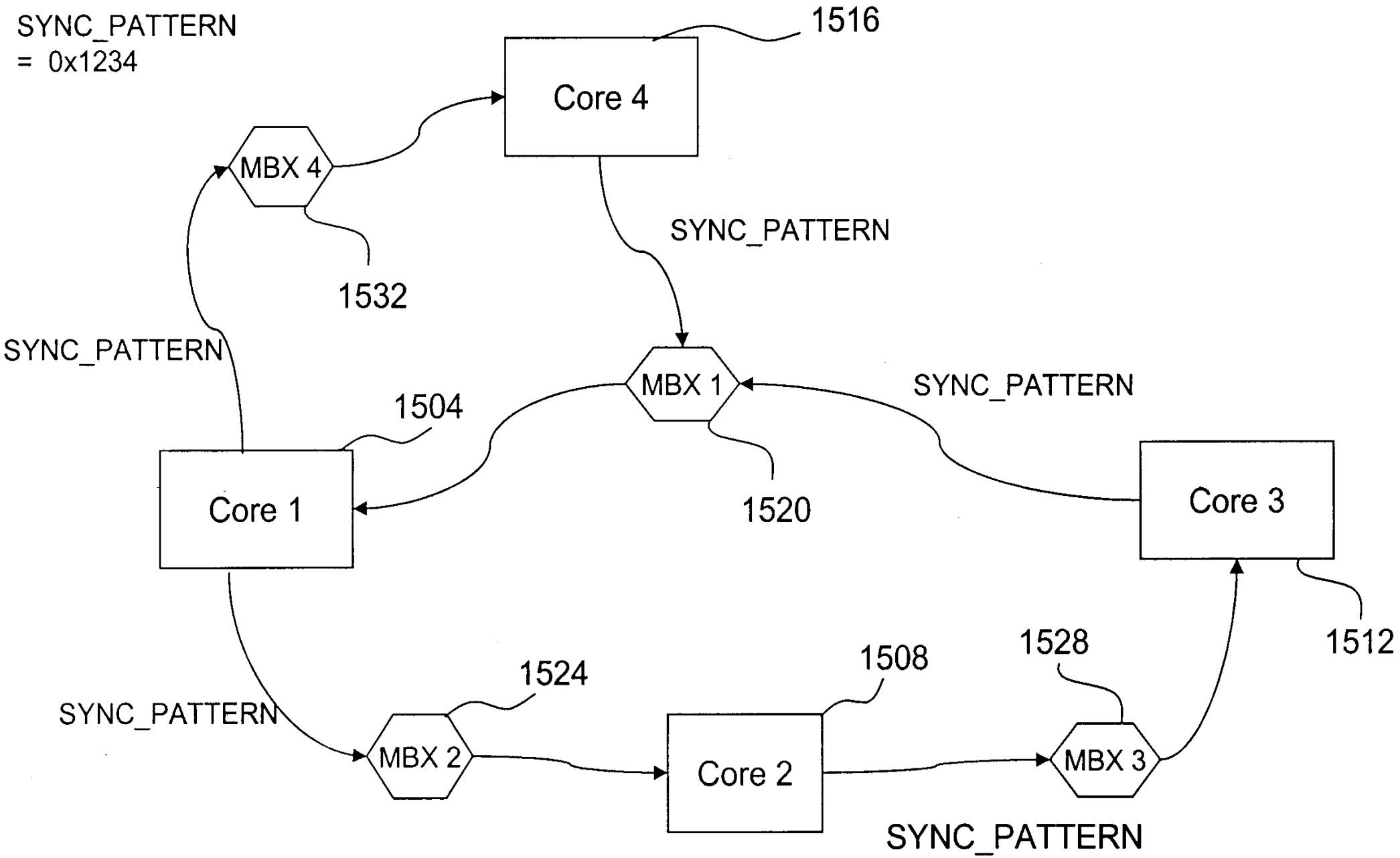


FIG. 15

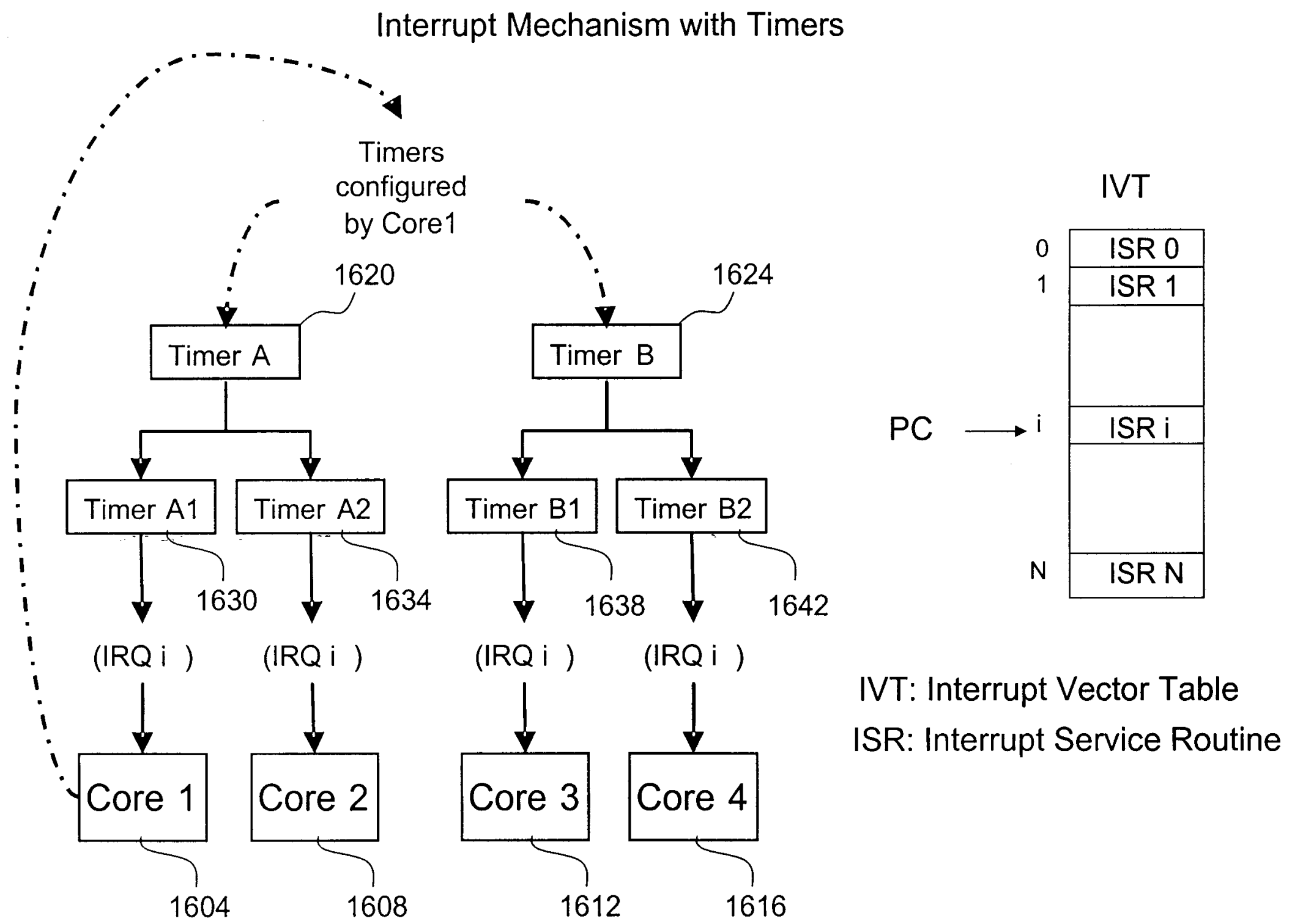
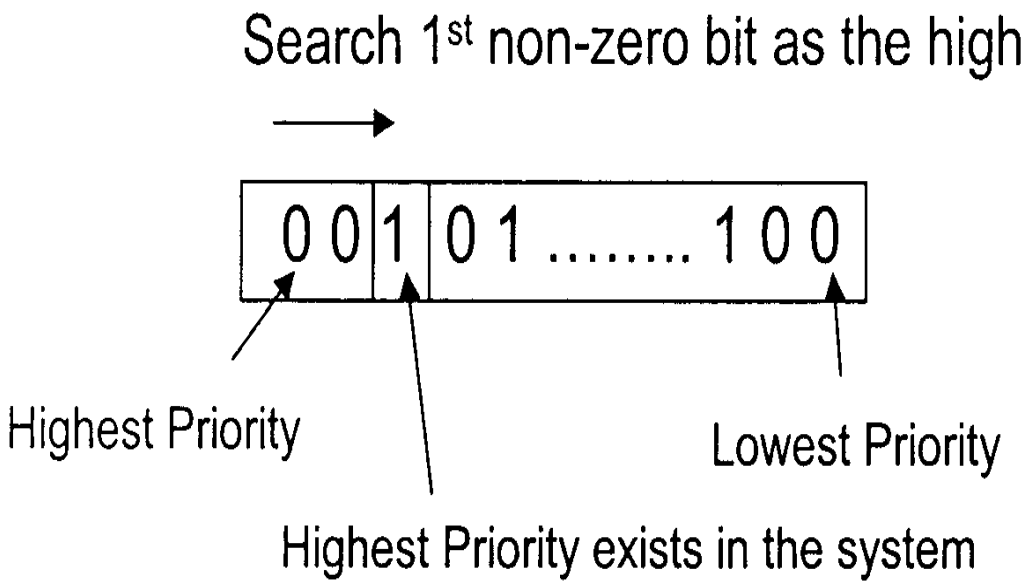

Interrupt Mechanism with Timers
Timers configured by Core1
1620
1624
Timer A
Timer B
Timer A1
Timer A2
Timer B1
Timer B2
1630
1634
1638
1642
(IRQ i)
(IRQ i)
(IRQ i)
(IRQ i)
Core 1
Core 2
Core 3
Core 4
1604
1608
1612
1616
IVT
0
ISR 0
1
ISR 1
PC
i
ISR i
N
ISR N
IVT: Interrupt Vector Table
ISR: Interrupt Service Routine

FIG. 16

Search 1st Non-Zero Bit

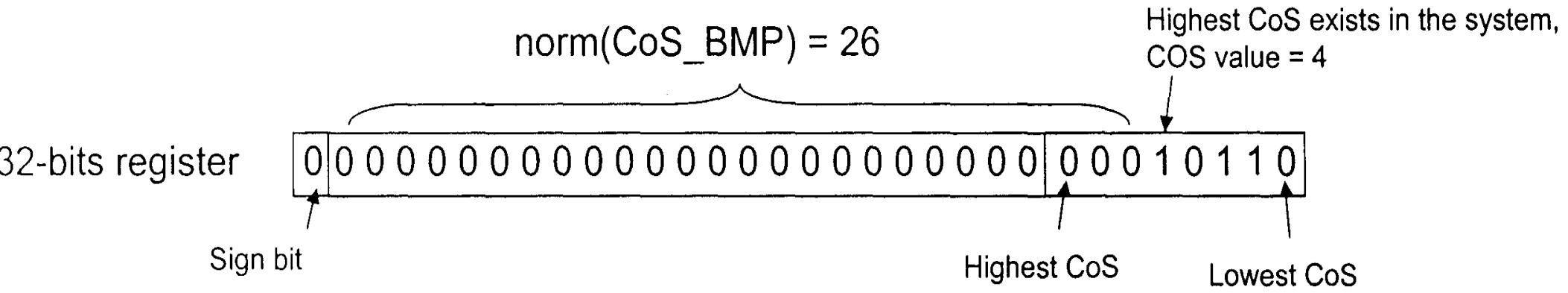


Priority: Can be

1. Class of Service (CoS)
2. TimeStamp value used to determine traffic (e.g., flow/packets) delivery sequence

FIG. 17

Search Highest CoS exists in the system using "**norm**"



norm(COS_BMP): Calculate how many redundant 0 from left to right (excludes the sign bit) in the COS_BMP

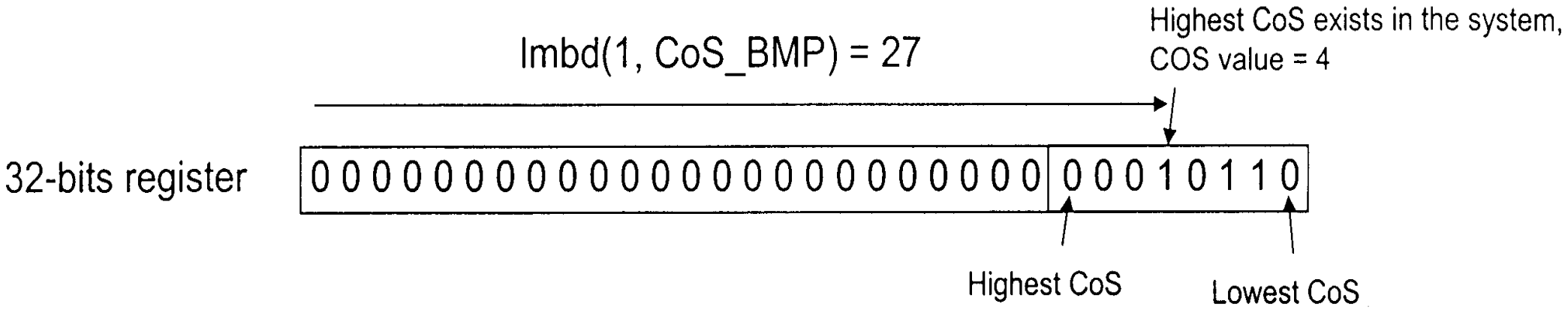
- A5 = CoS_BMP

Assembly Language		Meanings	Results
mvk	30, A6	A6 ←30	A6 ←30
norm	A5, A5	A5 ← norm(CoS_BMP)	A5 ← 26
Sub	A6, A5, A6	A6 ← A6 – A5	A6 ← 30 – 26

Highest CoS exists in the system = 4

FIG. 18

Search Highest CoS exists in the system using "**lmbd**"



lmbd(1, COS_BMP): Find the 1st "bit 1" position from left to right in the COS_BMP

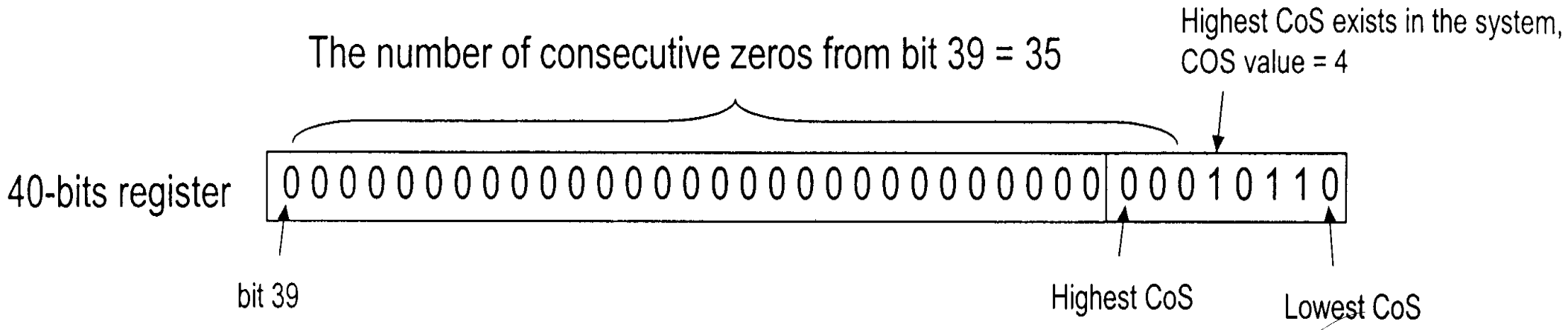
- A5 = CoS_BMP

Assembly Language		Meanings	Results
mvk	31, A6	A6 ←31	A6 ←31
lmbd	1, A5, A5	A5 ← lmbd(1, CoS_BMP)	A5 ← 27
Sub	A6, A5, A6	A6 ← A6 – A5	A6 ← 31 – 27

Highest CoS exists in the system = 4

FIG. 19

Search Highest CoS exists in the system using "**clb**"



clb(COS_BMP): use fix value ,9, to subtract the number of consecutive zeros from bit 39

- A5 = CoS_BMP

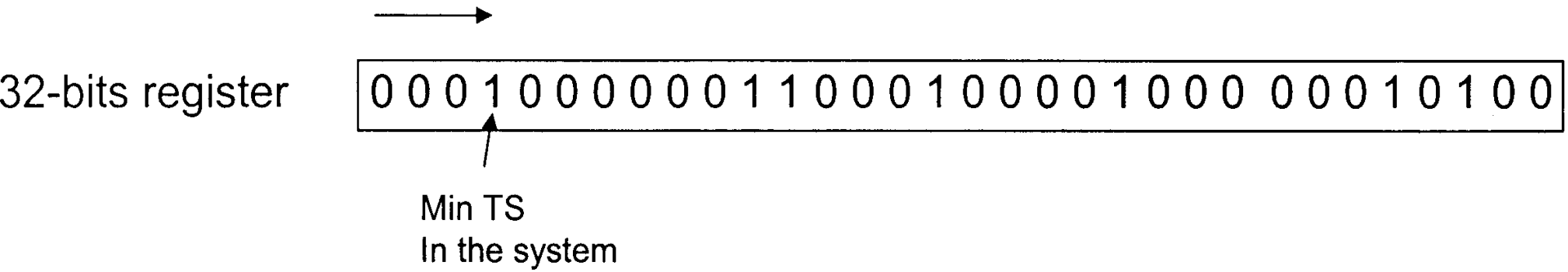
Assembly Language		Meanings	Results
clb	A5, A5	A5 ←clb(COS_BMP)	A5 ← 9 – 35
add	30, A5	A5 ← A5 +30	A5 ← – 26 + 30

Highest CoS exists in the system = 4

FIG. 20

Search Minimum TimeStamp value using "**lmbd**"

lmbd(1, TS_BMP) = 3



lmbd(1, TS_BMP): Find the 1st "bit 1" position from left to right in the TS_BMP

- A5 = TS_BMP

Assembly Language		Meanings	Results
lmbd	1, A5, A5	A5 ← lmbd(1, TS_BMP)	A5 ← 3

Minimum TimeStamp value = 3

FIG. 21

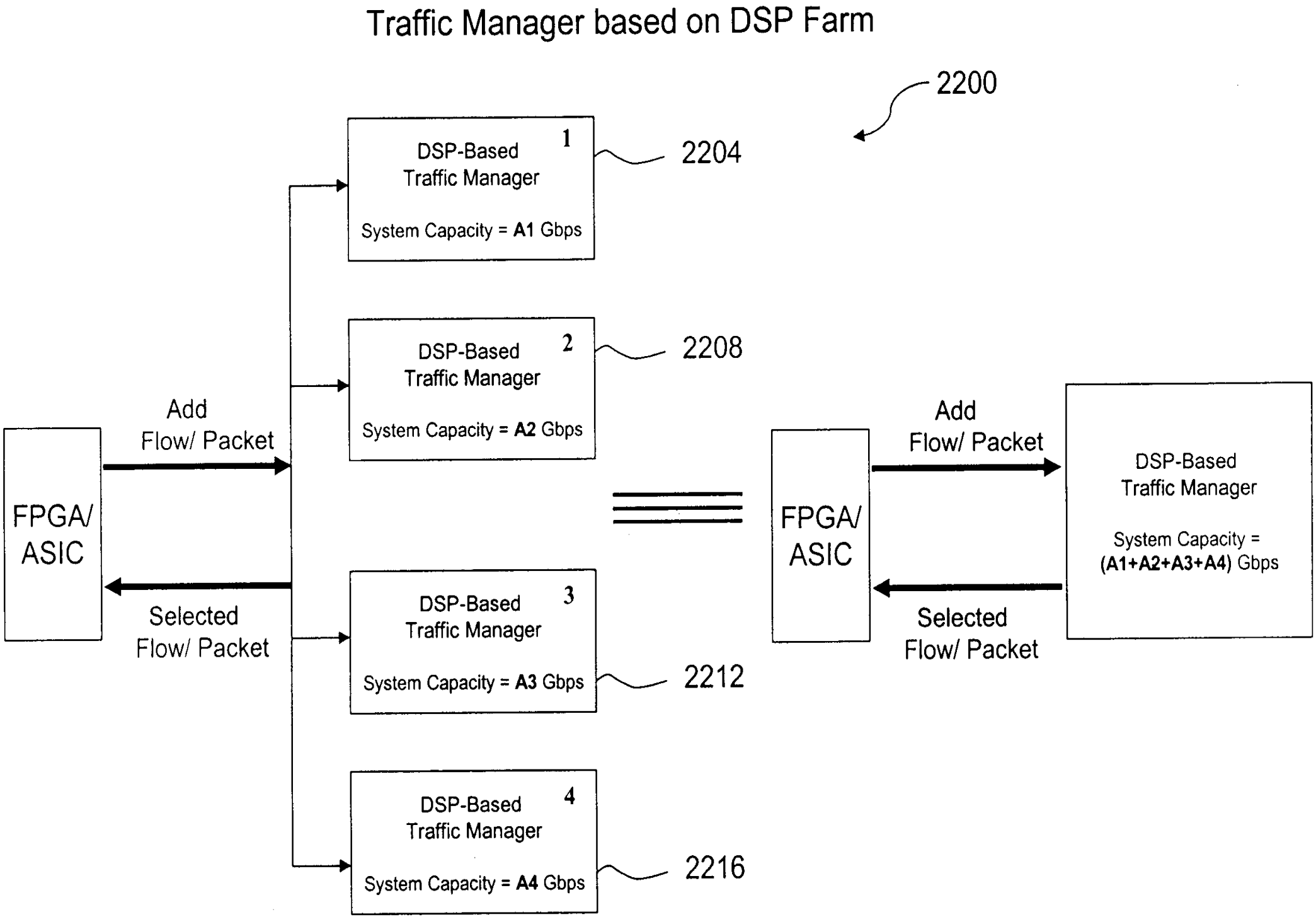

Traffic Manager based on DSP Farm
2200
DSP-Based Traffic Manager 1
System Capacity = A1 Gbps
2204
DSP-Based Traffic Manager 2
System Capacity = A2 Gbps
2208
DSP-Based Traffic Manager 3
System Capacity = A3 Gbps
2212
DSP-Based Traffic Manager 4
System Capacity = A4 Gbps
2216
FPGA/ ASIC
Add Flow/ Packet
Selected Flow/ Packet
FPGA/ ASIC
Add Flow/ Packet
Selected Flow/ Packet
DSP-Based Traffic Manager
System Capacity = (A1+A2+A3+A4) Gbps

FIG. 22